

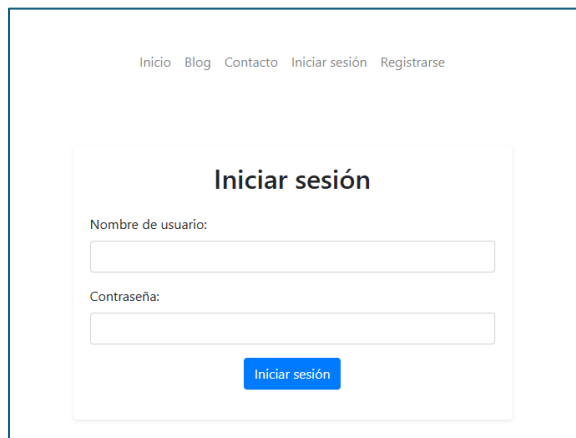
Informe del Ejercicio Práctico: Detección de Formularios HTML y Campos de Entrada

1. Objetivo del Ejercicio:

El objetivo principal de este ejercicio fue construir un script en Python que simule una fase inicial de reconocimiento web automatizado. El script debe conectarse a un sitio web, identificar y mostrar los formularios HTML presentes en la página, junto con sus respectivos campos de entrada. Esto permite al usuario observar estructuras web relevantes como parte de la superficie de ataque potencial, destacando el uso de librerías para solicitudes HTTP y análisis de HTML.

2. Descripción del Programa:

El script `detectar_formularios.py` es una herramienta de línea de comandos diseñada para realizar un análisis pasivo de páginas web. Permite al usuario especificar una URL y, a continuación, el programa se encarga de descargar el contenido HTML, parsearlo y extraer información detallada sobre cada formulario (`<form>`) y sus campos de entrada (`<input>`, `<textarea>`, `<select>`). La información incluye el método de envío (GET/POST), la URL de acción y los nombres y tipos de cada campo.



The screenshot shows a web page with a navigation bar at the top containing links: Inicio, Blog, Contacto, Iniciar sesión, and Registrarse. Below the navigation bar is a login form titled "Iniciar sesión". The form contains two input fields: "Nombre de usuario:" and "Contraseña:". Below these fields is a blue button labeled "Iniciar sesión".

```
Ingresa la URL a analizar (ej. https://example.com): https://blandskron.com/accounts/login/
--- Iniciando detección de formularios en: https://blandskron.com/accounts/login/ ---
Se encontraron 1 formulario(s).
```

```
-----
🔍 Formulario detectado #1:
Método: POST
Acción: https://blandskron.com/accounts/login/[sin acción definida]
Campos:
* name: csrfmiddlewaretoken | type: hidden
* name: username | type: text
* name: password | type: password
```

1. Funcionalidades Implementadas:

- **Conexión HTTP y Análisis HTML:**

- Utiliza la librería requests para realizar solicitudes HTTP GET a la URL proporcionada por el usuario.
- Emplea BeautifulSoup para parsear el contenido HTML de la respuesta, facilitando la navegación y extracción de elementos del DOM.
- Incluye un timeout de 5 segundos en las solicitudes para prevenir esperas indefinidas ante servidores lentos o no responsivos.

- **Detección y Extracción de Formularios:**

- Identifica y extrae todos los elementos <form> presentes en el HTML de la página.
- Los formularios detectados se numeran comenzando desde 1.

- **Extracción de Atributos del Formulario:**

- Para cada formulario, el script extrae y muestra:
 - El method de envío (GET o POST).
 - La URL de action (ruta de envío de los datos). La herramienta es capaz de resolver URLs de acción relativas a una URL absoluta, lo cual es crucial para la precisión del análisis.
 - Se proporcionan mensajes por defecto más claros (ej., [sin acción definida]) si los atributos no están presentes.

- **Identificación de Campos de Entrada:**

- El script localiza todos los campos de entrada relevantes dentro de cada formulario: <input>, <textarea> y <select>.
- Para cada campo, muestra su atributo name y su type (o el nombre de la etiqueta para textarea y select).
- La búsqueda de estos campos se ha consolidado en una sola operación find_all, haciendo el código más conciso y eficiente. Se utilizan mensajes por defecto como [sin nombre] para campos sin atributo name.

- **Manejo de Errores y Robustez:**

- Incorpora un manejo de excepciones exhaustivo para requests (errores de conexión, timeout, MissingSchema, HTTPError) y otras excepciones genéricas, haciendo el script más robusto frente a URLs inválidas o problemas de red.
- La URL de entrada del usuario se valida para asegurar que contenga un esquema (HTTP/HTTPS), añadiendo http:// por defecto si falta.

- **Consideraciones de Seguridad y Criticidad:**

- Cualquier formulario que recoja información sensible (credenciales, datos personales, financieros) o que permita modificar/eliminar datos en el servidor (ej., formularios de login, registro, comentarios, carga de archivos, etc.) sería crítico si no estuviera adecuadamente validado en el servidor, ya que podría ser susceptible a inyecciones (SQL, XSS), CSRF, o manipulación de datos.